The use of Artificial Intelligence in teaching: Bridge and other trick-taking card games
CPSC 490 Proposal
Hari Ganesan
Dana Angluin, Advisor

Background
Card games have been around for centuries, and possibly the most popular game of the last century is Bridge. Bridge is a trick-taking card game that requires four people in teams of two and a standard 52-card deck. Bridge is often played with a standard set of rules and small variations that encompass bidding systems and style of play. Unlike many other card games, Bridge requires a great amount of memory and logic.

Software for card games – and therefore, AI for card games – has been around for the last couple decades. The World Computer-Bridge Championship started in 1997, and there have been a couple AIs that have accounted for 11 of the 16 titles – Jack and WBridge5. Since Bridge is a game of incomplete information, these programs generate many samples of possible hands that other players might have, and then make decisions by weighing the various options differently.

Recently, there has there been a transition to web and mobile applications to support these games. However, my main interest remains standard game development, and along with it, engines able to support high-end graphics and sound. Therefore, I will be building this project using SDL, a popular game programming interface, and OpenGL, a standard graphics library, which I will then port to Direct X if time permits. All code will be written in C++.

Goals
One main facet of the AI in such games is to be able to teach people how to play the game. This will be my main goal of creating the application – to create an AI that walks a person through various stages of the game.

There are several parts to developing the game, and most involve the AI:

1) Setting up the structure for the main play (cards, score, legality)
2) Setting up the structure for bidding
3) AI: Bidding based on hand (using American standard)
4) AI: Keeping track of high cards and trumps
5) AI: Making decisions in trump
6) AI: Making decisions in notrump
7) Developing the teaching application
8) Making a menu and settings

In order to make the game as efficient as possible, AI decisions will be made based depending on “public” information available to the particular player – either based on his hand, the dummy hand, on previous cards played in the round, and on the bidding. Each player will have different sets of information available, and these sets must be understood in their full capacity in order to teach the application to the viewer.

The teaching application will present a UI that the user should find easy to navigate. There should be an instructions text box, as well as forward and back buttons so that one can go back and review instructions. A table of contents will also be included, so that one can review certain categories of learning the game at any particular time.

Deliverables

- Binary Files (one for Unix/Linux and one for Unix/OS X)
- Source Code, Makefile
- Notes
- Final Report

All material will be publically available on Github.

Future

There are a lot of directions where I could go after this project. There are many different bidding conventions, and I could make them in-game options. There are many trick-taking card games, such as Spades, Hearts, and Euchre, all of which could inherit my “CardGame” class and modify it accordingly. Finally, a possibility is to port the system over to Windows and use Direct X instead of OpenGL to run graphics.

References

SDL - http://www.libsdl.org/
OpenGL - http://www.opengl.org/